Web Services Proof-of-Concept

Test Plan--DRAFT

Version 1.0

07/01/2002

Table of Contents

Revision History

Date	Version	Author	Description
07-01-02	1.0		Initial Draft

1. Introduction

1.1 Purpose and Scope

This document specifies the objectives and approach for conducting test for the Web Services Work Group Proof-Of-Concept. A high-level test strategy and high level test case scenarios are outlined in this document. After testing, the document is updated with the test results, problem areas, and any information pertaining to rework, review, and retesting.

1.2 List of Reference Documents

(Describe all applicable reference documents. At a minimum, references to the high level software design, high level hardware or integration environment description, and a reference to problem/resolution documents)

	Document Name / Location	Document Description
1	VABC_BEA WS POC.doc	This proof of concept document outlines team member names,/roles, a high level design of the application, hardware and software requirements, and miscellaneous project comments.
2		
3		
4		
5		
6		
7		

2. Testing Approach and Objectives

2.1 Scope of Proof-Of-Concept Tests

The tests will focus on the functionality and implementation of key web services features. The limited business logic that utilizes the web service features will also be tested but with less emphasis. The tests will review data flows between the application and the web services; as well as, monitoring for any potential problems in performance or bottlenecks. Detailed results of all of the tests will be compiled, reviewed, and all appropriate documentation will be updated. Several test cycles are anticipated and testing will continue until satisfactory results are achieved.

3. Individual Steps and Test Descriptions

3.1 System Wide Functional Test Descriptions

3.1.1 Tested Features

Scenario / Test Case Name	Test Case Document(s)
Test get Address functionality via services	
Test methods associated with changing or updating an address	
Test queryAddress functionality via services.	
Test ability to register services in the UDDI.	
Test client look up of UDDI Services.	
Test invokes a service on the WSDL from the UDDI.	
Test performance on a queryAddress functionality using the services.	
Test Another Agency for changing an address	
Validation test of Agency Logs.	

3.1.1.1 Test gets Address functionality via services.

The JUnit test will call the getAddress WebService to retrieve a known address from the VABC test database.

Prerequisites: Have a known address in the test database.

- Statically bind to the WSDL for the getAddress WebService
- Invoke the getAddress WebService with the known AddressID
- Retrieve the getAddress data and write it to a log for validation.

3.1.1.2 Test methods associated with changing or updating an address

The JUnit test will call the getAddress WebService to retrieve a known address from the VABC test database.

Prerequisites: getAddress must function; Have a known address in the test database.

- Statically bind to the WSDL for the changeAddress WebService
- Invoke the changeAddress WebService with the known AddressID, but with different street and city info
- Retrieve the same address with getAddress data and write it to a log for validation. The data should have changed
- Invoke the changeAddress WebService with a new address.
- Retrieve the same address with getAddress data and write it to a log for validation. The data should be there.

3.1.1.3 Test queryAddress functionality via services.

The Unit test will call the queryAddress service with a data range to receive a known set of address from the log.

Prerequisites: A known set of data is in the test database to validate the correct return addresses.

- Statically bind to the WSDL for the queryAddress WebService
- Invoke the queryAddress service with a defined date range and address type.
- Retrieve the set off addresses
- Write the address to a log for validation
- Repeat the process for another address type and alternate date ranges.
- Run the test for changeAddress to insert a new address
- Repeat the test for the current date and address type as used in changeAddress
- Validate that the addresses updated and added in changeAddress are in the log query results.
- Write the results to a log file for validation.

3.1.1.4 Test ability to register services in the UDDI.

Client software must be able to publish to the UDDI register and set properties in the Category Bag to identify the agency name, service, type of addresses to receive, and the WSDL URL for the service to provide the address get, change and query operations.

A JUnit test will be written that will do the following

- Publish a set of address services to the UDDI registry,
- Search for these same services.
- Write to a log file for validation that the services were in fact registered.
- Delete the test services from the UDDI registry

3.1.1.5 Test client look up of UDDI Services.

The client software must be able to search the UDDI registry for a given address type, retrieve the WSDL URL. This test will be executed and validated as the search portion of the register services to UDDI test.

3.1.1.6 Test invokes a service on the WSDL from the UDDI.

The service must be able to retrieve a WSDL from the UDDI search and with this WSDL, Bind to the web service, invoke the WebService and return the results of the WebService.

This test will require that the getAddress functionality and the UDDI Search functionality is already functioning. The test will do the following:

Prerequisites: UDDI Search works, getAddress WS works.

- Publish a UDDI entry for the getAddress service. Set one of the categoryBag properties to “VABCTest”.

- Search for the VABCTest getAddress service in the UDDI
- Retrieve the VABCTest getAddress WSDL
- Bind to the WSDL end point
- Invoke the getAddress service and retrieve a known address in the VABC site.
- Write the retrieved address to a log for validation that the test completed correctly.
- Delete the VABCTest getAddress service from the UDDI

3.1.1.7 Test performance on a queryAddress functionality using the services.

Write a JUnit test to retrieve addresses from QueryAddress with increasing date ranges. The objective of the test will be to determine a performance curve for the number of addresses returned.

3.1.1.8 Test Another Agency for changing an address.

Create a JUnit test to test that address changes propagated to another agency did in fact occur.

Caution: This test can only be executed in a controlled test environment. Potential side effects in target agency data.

Prerequisite: Another agency has subscribed for a business address.

- Create a dummy Address for Testing purposes

Name=VABCDUMMY Address
AddressID=99955999
Street1=street (N)
City=Richmond

- Search the UDDI registry for an agency accepting business address
- Invoke a getAddress on the target agency for the AddressID of our dummy address.
- If the address exists, validated that the name is “VABCDUMMY Address” if not decrement the AddressID and try again.
- If not found invoke changeAddress to insert the address.
- If found increment the Street (N) number and invoke changeAddress to update the address
- Invoke getAddress to retrieve the “VABCDUMMY Address”
- Validate that the Street1 equals the update or inserted value. Write results to a log
- Write the results to a Log file for Validation
- Repeat the test on the next agency subscribing to business addresses.

3.1.1.9 Validation test of Agency Logs.

Validate Log entries between agencies.

- VABC gets an address change for a Business address.
- VABC looks into the UDDI to get the list of agencies that have subscribed for Business Addresses e.g. VRS, VMS.
- VABC makes an entry into their log of the address data sent, with the Source being VABC.
- VABC sends the address via the changeAddress method to VRS and VMS

- VRS and VMS receive the address and record the entry in their respective logs. With the source being VABC
- During Validation the VABC log is interrogated and the entry is found with a source VABC (source=agency Log owner means a sent address). Then the agencies registered for Business addresses logs are checked to validate that they received the given address with the VABC sender.

This process will validate that addresses sent were actually received by the subscribing agencies.

3.1.2 Discussion / Notes

4. Tools and Test Equipment Required

(Identify all tools and test equipment needed to accomplish the integration. Integration protocols will specify in detail all configuration information, set-up procedures, parameters, environmental conditions (lighting, humidity, etc.), etc. necessary for the integration testing. Summarize any significant configuration efforts, special environmental requirements, etc.)

4.1 Key Dependencies

(Define key dependencies in the schedule (people or equipment resources, availability of other hardware or software under development, etc.).)

-
-

4.2 Risks and Assumptions

-
-

5. Problem Recording and Resolution

(Define the mechanism to be used for problem recording and resolution, including any necessary rework of documents, software or hardware elements, test plan or procedures. Include (or reference) any forms. Reference any standard problem tracking procedures that will be used.)

The VABC/BEA Team will compile problems and Issues that either suspend testing or require a restart of testing into a separate document for review.

6. Rework, Review, and Retest Procedures

(Define the process for rework, review, and retest of any element needing modification. This process must include sufficient retest to verify that any modifications have not impacted other functions already tested. This process must also define the project guidelines for how related design and requirements documents will be updated as changes are made. Change management processes for this phase must be defined and followed.)

Upon completion of testing and upon update of all test documentation, a review will be held between the VABC/ BEA Team to discuss test results any rework, review, retest, restart or suspensions. This information will be documented, along with supporting test results and will be made available to the workgroup.

7. Suspension, Restart, and Exit Criteria

(Define criteria for suspending testing before completion (for instance, the discovery of major problems with a certain feature), criteria for restarting testing following such a suspension, and the criteria for determining that integration test is complete and system test can begin

System tests between will be suspended and cannot continue if the following conditions occur:

- N/A

8. Success Criteria

Tests will run until such time that all test case scenarios are completed successfully. (This section to be updated further…)